

SMART CONTRACTS

WHAT DID I LEARN ABOUT ETHEREUM SMART CONTRACTS?



Gesiel Rosa
@gesielrosa

ROADMAP

- Blockchain basics
- Ethereum Virtual Machine
- Smart Contract basics
- Solidity
- Compile and deploy
- Interaction (DApp)

BLOCKCHAIN

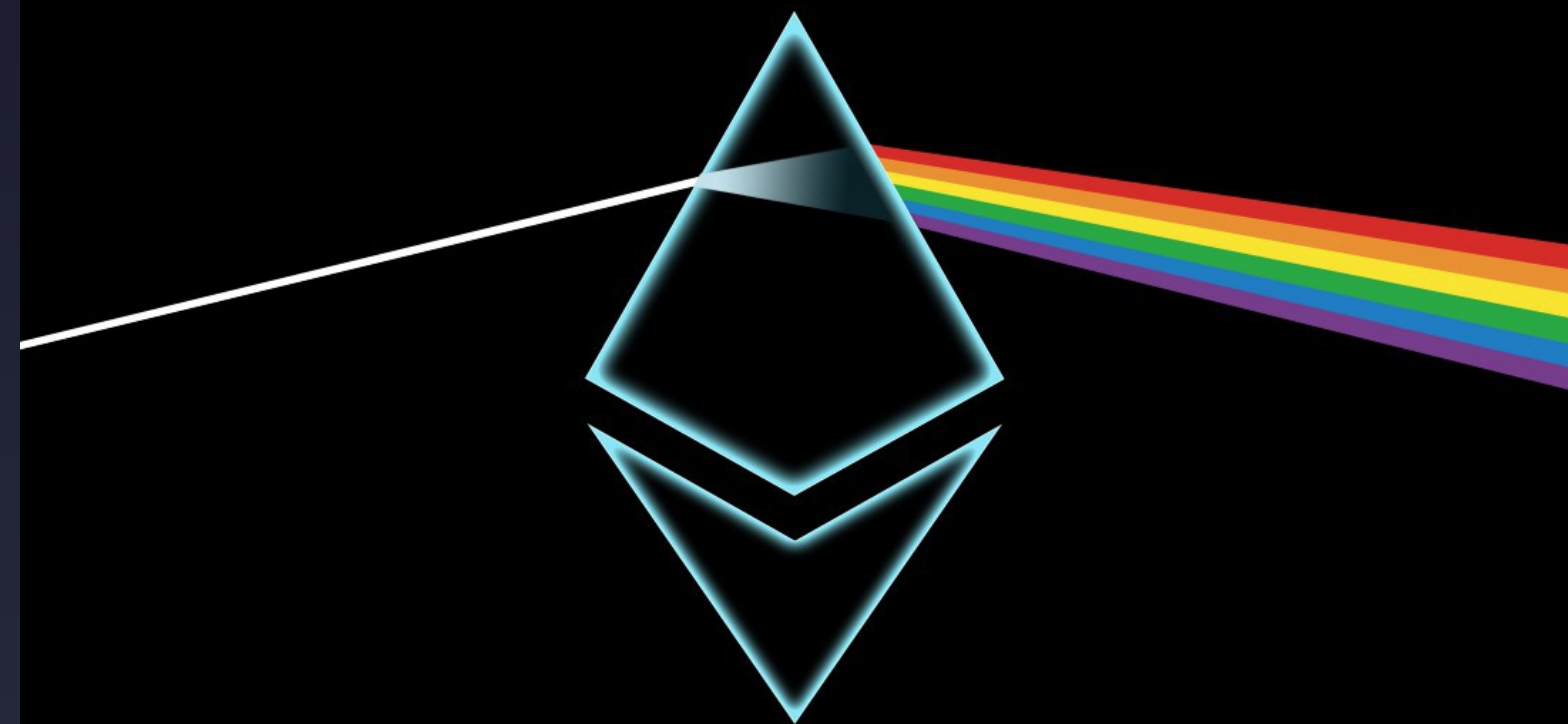


BLOCKCHAIN

- Transactions
- Blocks

EVM

ETHEREUM VIRTUAL MACHINE



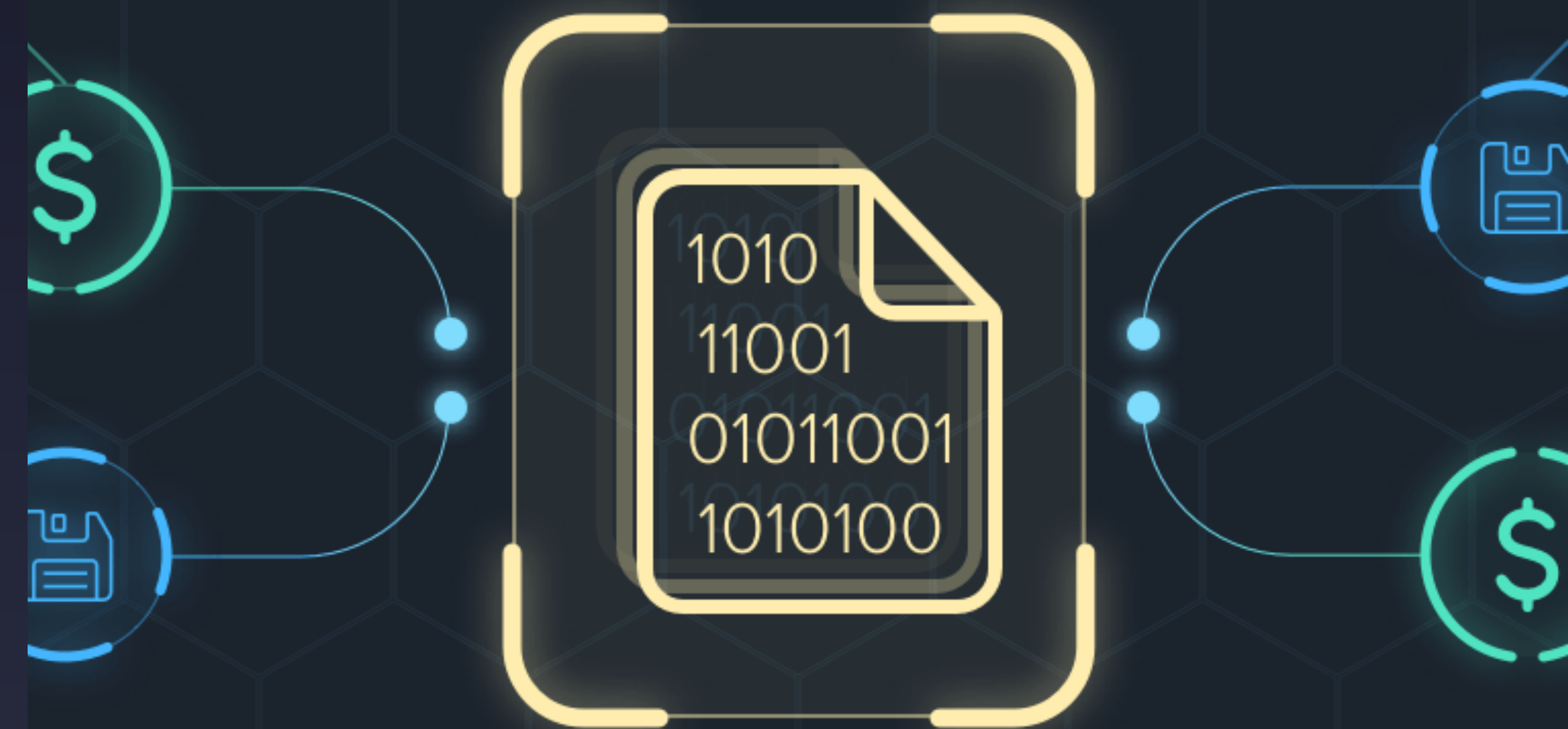
ethereum

ETHEREUM VIRTUAL MACHINE

- Runtime environment for smart contracts
- Accounts
 - External - free
 - Contract - paid
- Transactions
- Gas

SMART CONTRACTS

Smart contract



SMART CONTRACTS

- Account with codes
- Enable on blockchain
- Transaction changes account data
- DApp

SOLIDITY

```

allowance(address spender, uint256 addedValue) public virtual returns (bool) {
    _approve(msgSender(), spender, _allowances[msgSender()][spender].add(addedValue));
}

// Decreases the allowance granted to `spender` by the caller.
// This is an alternative to {approve} that can be used as a mitigation for problems
// described in {IERC20-approve}.
// Emits an {Approval} event indicating the updated allowance.
//
// Requirements:
// - `spender` cannot be the zero address.
allowance(address spender, uint256 subtractedValue) public virtual returns (bool) {
    _approve(msgSender(), spender, _allowances[msgSender()][spender].sub(subtractedValue, "ERC20: transfer amount exceeds balance"));
}

// Transfers `amount` from `sender` to `recipient`.
// Emits a {Transfer} event.
//
// Requirements:
// - `sender` cannot be the zero address.
// - `recipient` cannot be the zero address.
// - The caller must have enough balance to transfer this amount.
// - `amount` must be non-zero.
transfer(address sender, address recipient, uint256 amount) internal virtual {
    if (sender == address(0)) revert("ERC20: transfer from the zero address");
    if (recipient == address(0)) revert("ERC20: transfer to the zero address");
    _transfer(sender, recipient, amount);
}

// Transfers `amount` from `sender` to `recipient` and mints to `account`.
// Emits a {Transfer} event and a {Mint} event.
//
// Requirements:
// - `sender` cannot be the zero address.
// - `recipient` cannot be the zero address.
// - `account` cannot be the zero address.
// - The caller must have enough balance to transfer this amount.
// - `amount` must be non-zero.
transfer(address sender, address recipient, address account, uint256 amount) internal virtual {
    if (sender == address(0)) revert("ERC20: transfer from the zero address");
    if (recipient == address(0)) revert("ERC20: transfer to the zero address");
    if (account == address(0)) revert("ERC20: mint to the zero address");
    _transfer(sender, recipient, amount);
    _mint(account, amount);
}

```

SOLIDITY

- Influenced by C ++, Python and JavaScript
- Technical characteristics
 - Statically typed
 - supports inheritance
 - supports libraries
 - supports complex user-defined type

CODE



```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >=0.7.0 <0.9.0;

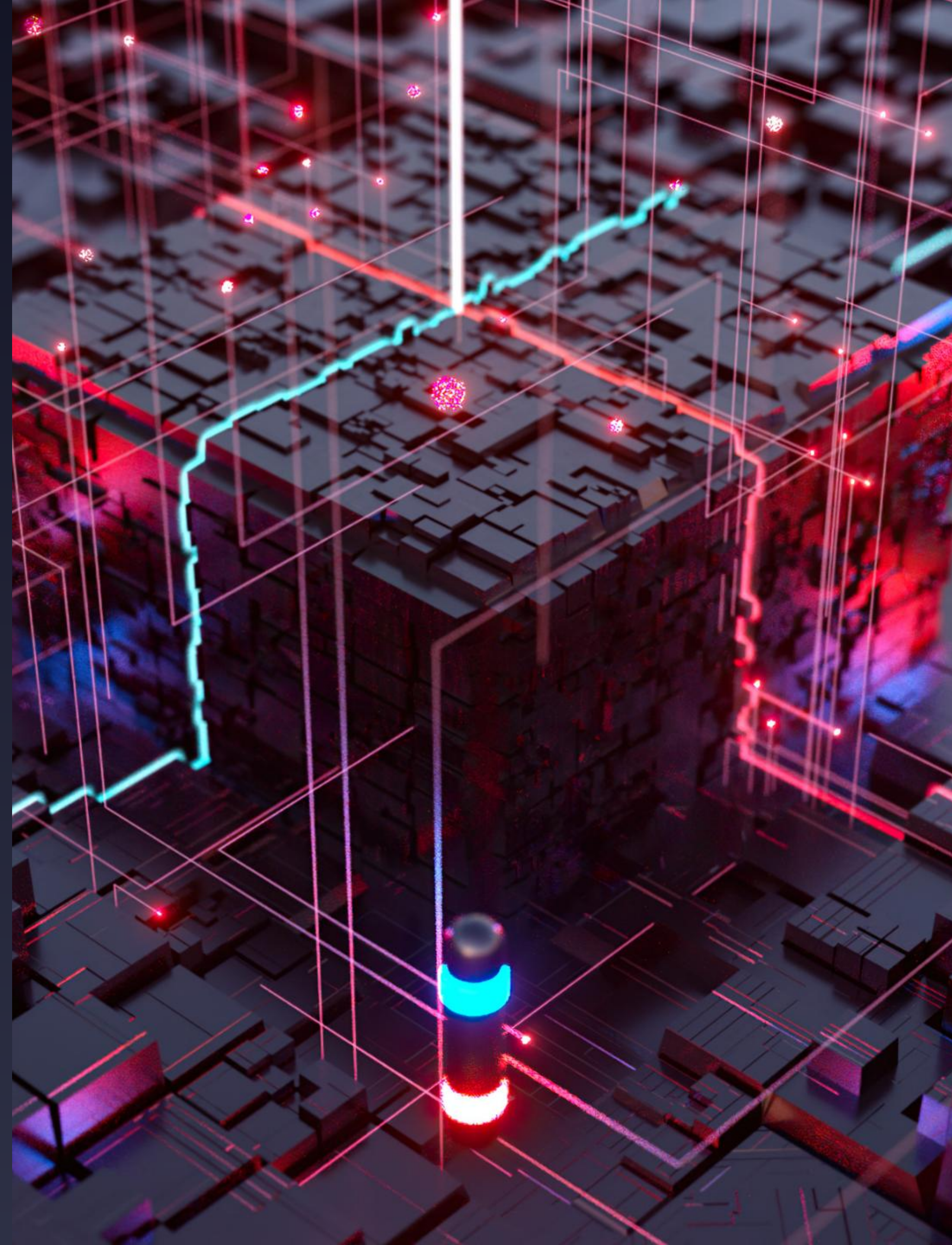
contract Storage {

    uint256 number;

    function store(uint256 num) public {
        number = num;
    }

    function retrieve() public view returns (uint256){
        return number;
    }
}
```


COMPILE AND DEPLOY



COMPILE

- Compiler: solc
 - NodeJS + solcJS
- Truffle Suite
- Input: contract.sol (solidity source code)
- Output: contract.json
 - Bytecode, ABI

DEPLOY

- contract.json
- Network (mainnet or testnet)
 - Ganache
- Account (external)
- Truffle Suite (again)
- Return: contract address

DAPP

DECENTRALIZED APPLICATIONS



DAPP

- Contract ABI and Address
- web3.js
 - Accounts, contracts
- MetaMask (provider/wallet)
- Contract instance
- Interact!!!

EXAMPLE DAPP

[HTTPS://SC.RHIZOM.DEV/](https://sc.rhizom.dev/)

- Interact with any Ethereum contract
- Angular + ~~@truffle/contract~~ + web3.js
- MetaMaskService
- ContractService
- SmartContract class

INTERACTING



```
const contract = new SmartContract<IContract>(injector, ABI, address);

contract.init();

contract.instance.MyFunction()
    .then(result => {
        console.log(result);
    })
    .catch(err => {
        console.log(err);
    });
```

THANKS

